```java
/*
 * Written by JJ Shepherd
 */
public class MaxHeap <T extends Comparable<T>>
{
    private T[] heap;
    private int size;//First null element
    public static final int DEF_SIZE = 128;

    public MaxHeap()
    {
        init(DEF_SIZE);
    }
    public MaxHeap(int aSize)
    {
        init(aSize);
    }
    private void init(int aSize)
    {
        if(aSize <= 0)
        {
            init(DEF_SIZE);
            return;
        }
        heap = (T[])(new Comparable[aSize]);
        size = 0;
    }
    public void add(T aData)
    {
        if(heap[heap.length-1] != null)
            return;
        heap[size] = aData;
        bubbleUp();
        size++;
    }
    private void bubbleUp()
    {
        int index = size;
        while(index > 0 && heap[(index-1)/2].compareTo(heap[index])<0)
        {
            T temp = heap[(index-1)/2];
            heap[(index-1)/2] = heap[index];
            heap[index] = temp;
            index = (index-1)/2;
        }
    }
    public T remove()
    {
        if(size <= 0)
            return null;
        T ret = heap[0];
        heap[0] = heap[size-1];
        heap[size-1] = null;
        size--;
        bubbleDown();
        return ret;
    }
    private void bubbleDown()
    {
        int index = 0;
        while(index*2+1 < size)
        {
            int bigIndex = index*2+1;
            if(index*2+2 < size && heap[index*2+1].compareTo(heap[index*2+2])<0)
                bigIndex = index*2+2;
            if(heap[index].compareTo(heap[bigIndex])<0)
            {
                T temp = heap[index];
                heap[index] = heap[bigIndex];
                heap[bigIndex] = temp;
            }
            else
                break;
            index = bigIndex;
        }
    }
    public T peek()
    {
        return heap[0];
    }
    public void print()
    {
        for(int i=0;i<size;i++)
            System.out.println(heap[i]);
    }
}

```